ASSIGNMENT 1

ITRI 626

ENRICO DREYER
31210783

Table of Contents

Table Of Figures

Introduction

In this assignment I learned about simplest form of learning, which is hot and cold learning. After making a prediction, you predict again with one higher weight and one lower weight. You then move to the one with the lowest error. This will result in an error that is close to 0.

A snippet of code was given to us, where we had to determine the step_amount and number_of_iterations to predict a value that is closest to the goal_prediction that is 1.

What I did

I googled Hyperparameter Optimization as suggested in class. This helped me understand what the assignment was about. I also found the website https://realpython.com/python-ai-neural-network/, this really helped me grasp on what to do in the assignment.

I read that calculating the derivative (2 * prediction – goal_prediction) you get to see if your step_amount should increase or decrease. As shown in the screenshot bellow, I started with a step_amount of 0.01 and the number_of_iterations of 4.

I also commented out the “print(“Error:” + str(error) + “ Prediction:” + str(prediction))” to make it easier for me to get the derivative.

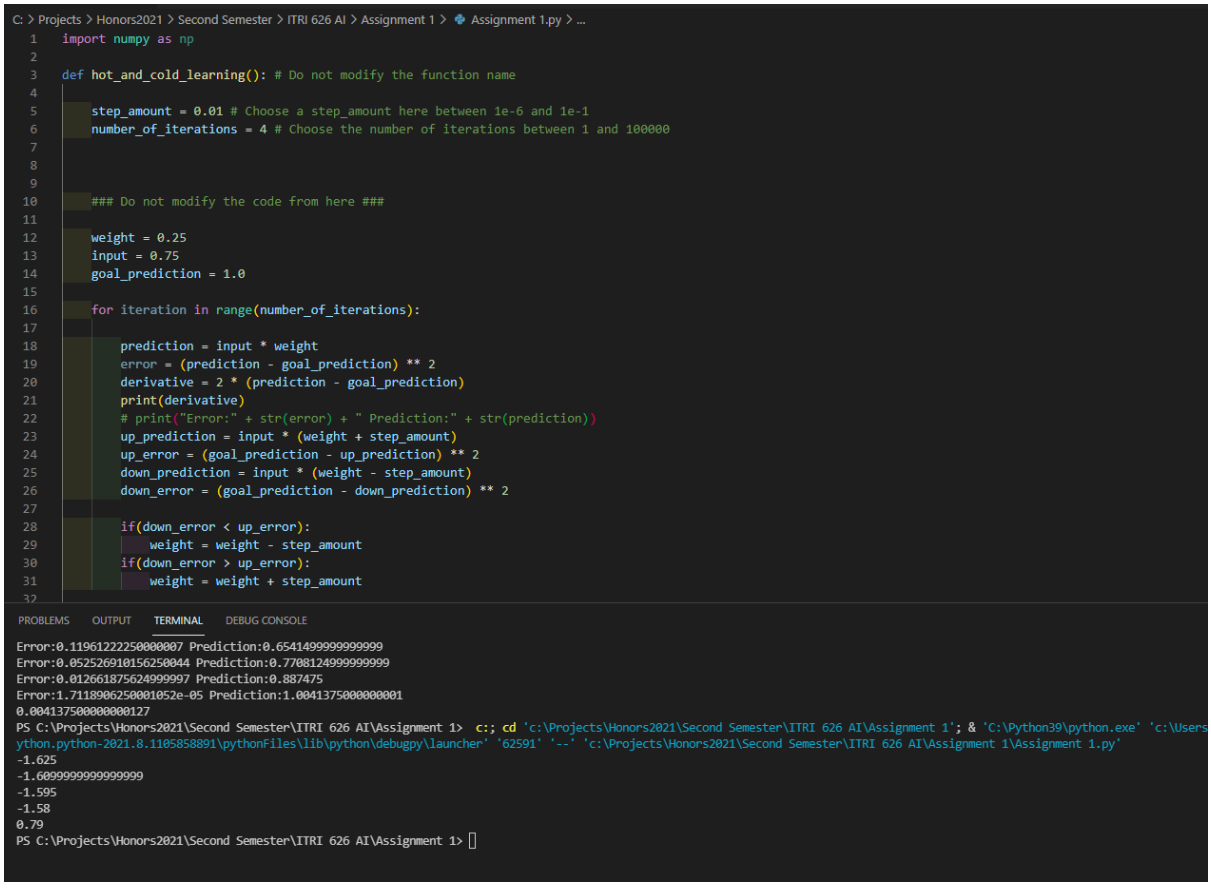
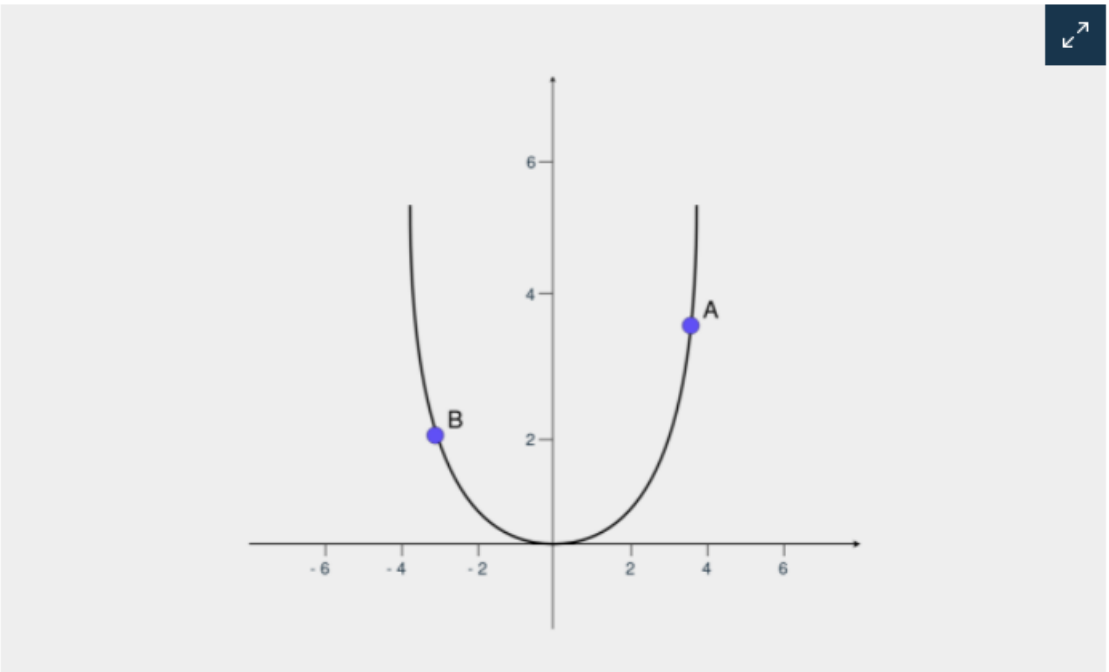


Figure 1: Getting the derivatives

According to (Mesquita, 2021), these four amounts showed me that I had to increase my step amount. I also made the number_of_iterations 25 at random.

This is based on a plot of a quadratic function.



Plot of a quadratic function

The error is given by the y-axis. If you're in point A and want to reduce the error toward 0, then you need to bring the x value down. On the other hand, if you're in point B and want to reduce the error, then you need to bring the x value up. To know which direction you should go to reduce the error, you'll use the **derivative**. A derivative explains exactly how a pattern will change.

Figure 2: Plot of a quadratic function (Mesquita, 2021)

After making my step amount 0.1, I realized that after my 8th iterations it started giving me the same two values in repeat, as shown in the screenshot below, and from what I could understand from Steven Tartakovsky (2017) this means that the graph has reached minimal optimal point, thus the prediction and error begin a loop of repetition between two points. Thus, making my number_of_iterations 8.

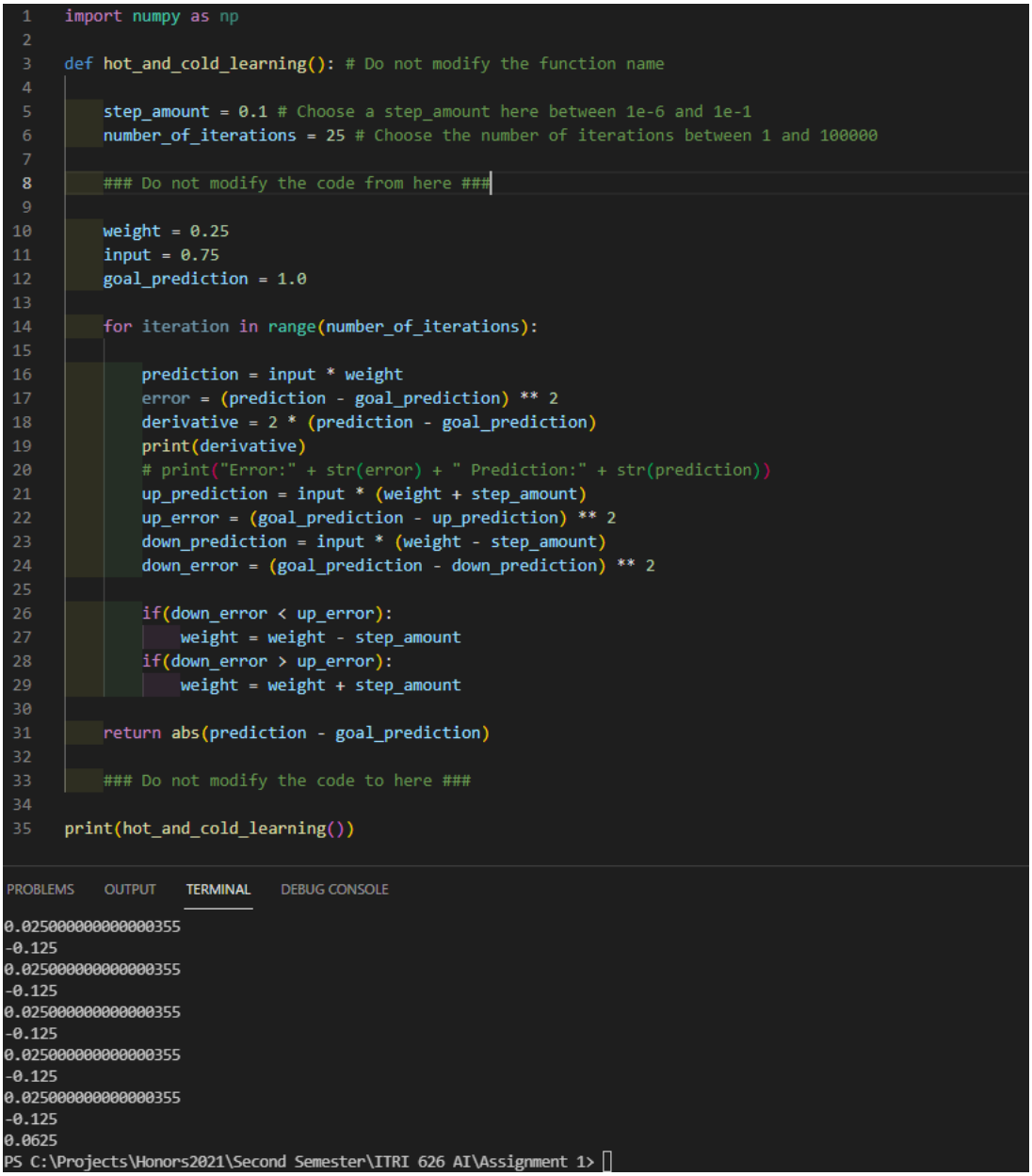


Figure 3: Getting repetitions of predictions

After changing the number_of_iterations to 8 I started playing around with the step_amount, while having the plot of a quadratic function in my head. I also tried using numpy for the graphs but felt more comfortable drawing it out.
My drawings on a white board is shown in the screenshot below.

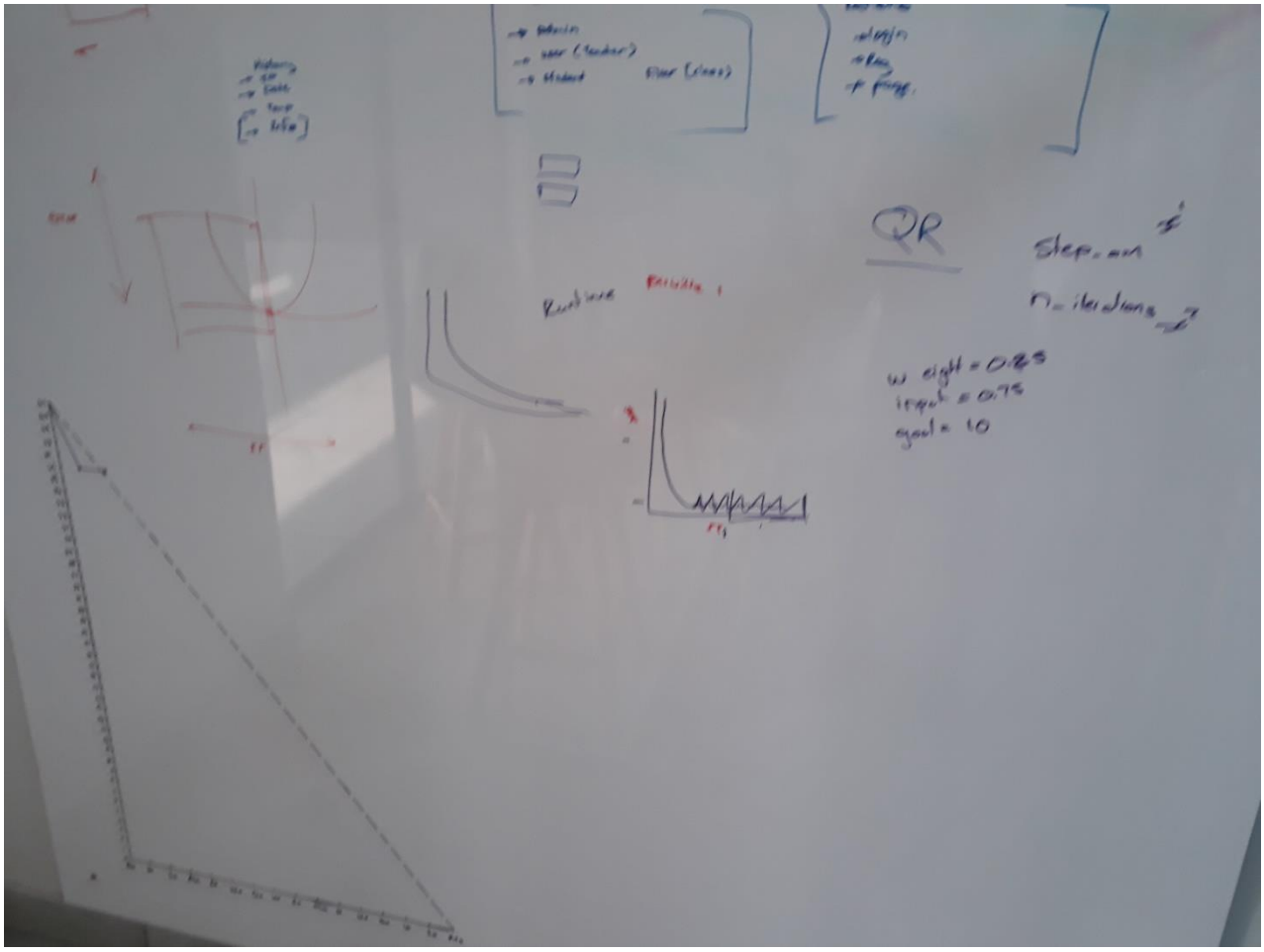


Figure 4: Drawings on board

Outcome

After playing around I figured out that the best outcome was a step_amount of 0.15555 and a number_of_iterations of 8.

This gave me an error of 1.7118906250001052e-05 and a prediction of 1.0041375000000001 with a result of 0.004137500000000127. As show in the screenshot below.

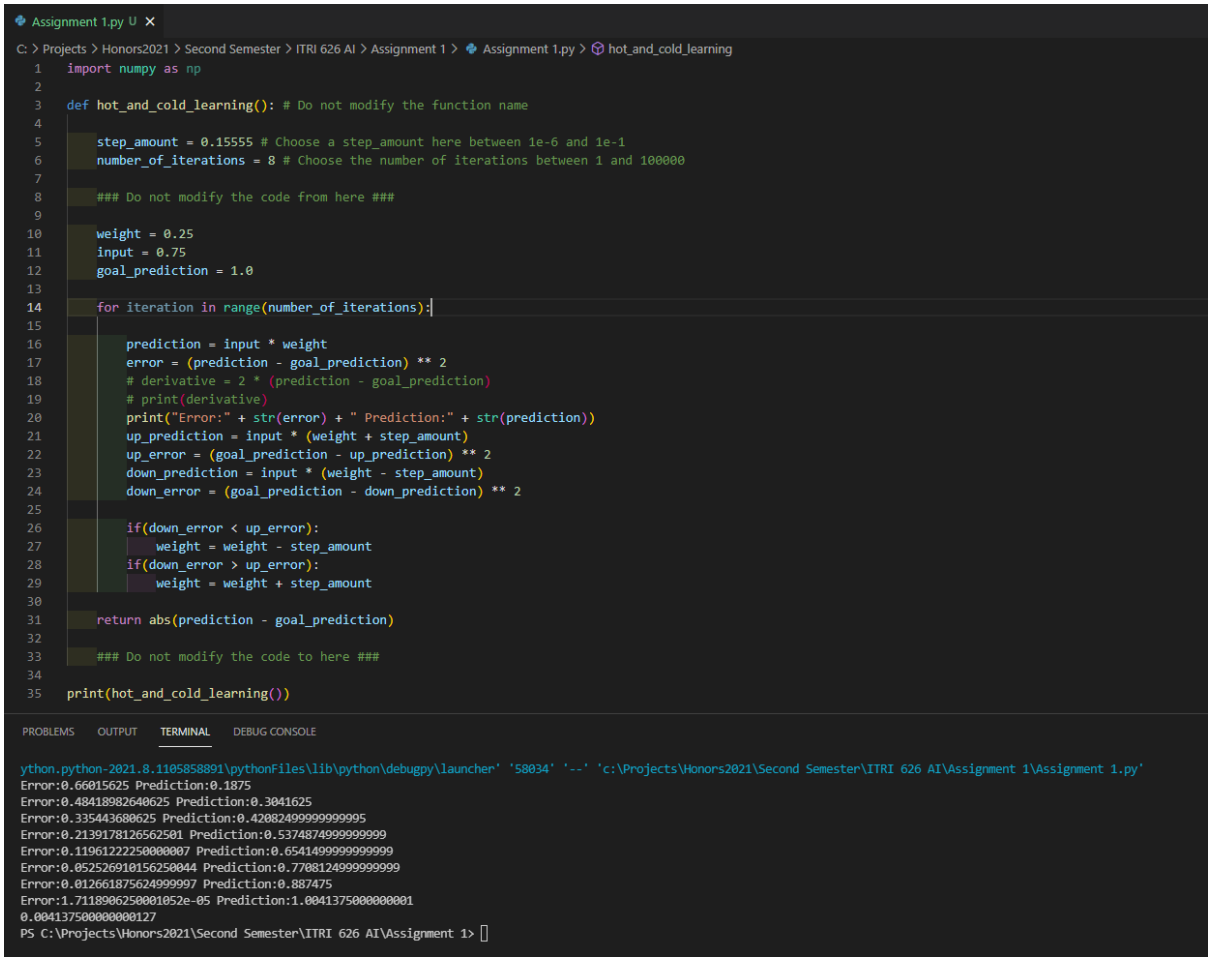


Figure 5: Final outcome

References

Mesquita, D. (2021). *Python AI: How to Build a Neural Network & Make Predictions*. https://realpython.com/python-ai-neural-network/

Steven Tartakovsky, S. C. a. M. M. (2017). *Deep Learning Hyperparameter Optimization with Competing Objectives*. https://developer.nvidia.com/blog/sigopt-deep-learning-hyperparameter-optimization/